Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tomaž Čufer

Poenostavitev ETL procesa z uporabo platforme Talend

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Dejan Lavbič

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Obvladovanje velike količine informacij je vedno večji izziv, saj je cilj večine organizacij čim bolje izkoristiti te podatke in na podlagi njih ukrepati za dosego večje konkurenčne prednosti. Trenutne raziskave se ukvarjajo predvsem z metodami za analizo te velike količine podatkov, zavedati pa se je potrebno, da je večji del napora potreben predvsem v fazi zbiranja, čiščenja in integracije podatkov iz različnih virov. V okviru diplomske naloge je potrebno raziskati področje avtomatiziranega pridobivanja, preoblikovanja in nalaganja velike količine podatkov iz različnih virov in pregledati pristope za reševanje tega problema. Preglejte najbolj pogosto uporabljana orodja na tem področju, ki jih kritično ovrednotite. Na podlagi izbranih scenarijev, predvsem s področja pridobivanja delno strukturiranih podatkov, prikažite prednosti in slabosti izbranega orodja.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Tomaž Čufer sem avtor diplomskega dela z naslovom:

Poenostavitev ETL procesa z uporabo platforme Talend

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Dejana Lavbiča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 16. marca 2015 Podpis avtorja:

Kazalo

Povzetek

Abstract

1 Uvod **1**

2 Podatkovno skladišče in ETL proces **3**

2.1 Pridobivanje (*Extract*) 3

2.2 Preoblikovanje (*Transformation*) 4

2.3 Nalaganje (*Load*) 4

2.4 Podatkovno skladišče 4

2.5 Razširjena ETL orodja 6

3 Talend **11**

3.1 Produkti 11

3.2 Delovno okolje 13

4 Reševanje klasičnih problemov obvladovanja podatkov s platformo Talend **15**

4.1 Uporabljena programska oprema 15

4.2 Integracija in skladiščenje izluščenih podatkov o podjetjih . . . 16

4.3 Zajemanje, obdelava in shranjevanje odzivov s socialnega omrežja Twitter 23

4.4 Primerjava priljubljenosti objav na Pinterestu z obiskanostjo izvirnih spletnih strani vsebine objav 28

4.5 Komponente 31

5 Zaključek **35**

5.1 SWOT analiza 35

5.2 Sklepne ugotovitve 35

Literatura **39**

Seznam uporabljenih kratic in izrazov

API (Application Programming Interface) Aplikacijski programski vmesnik.
ETL (Extract, Transform, Load) Pridobivanje, preoblikovanje in nalaganje
HTML (Hyper Text Markup Language) Označevalni jezik za izdelavo spletnih strani
SWOT (Strenghts, Weaknesess, Opportunities, Threats) Prednosti, slabosti, priložnosti, nevarnosti
tweet - Sporočilo, ki ga objavi uporabnik družabnega omrežja Twitter
retweet - Tweet uporabnika, ki ga ponovno objavi en ali več uporabnikov
hashtag - znak #, s katerim uporabniki označujejo tweete, ki se nanašajo na določeno temo
pin - Slika ali videoposnetek, ki ga uporabnik aplikacije Pinterest doda v svoj profil
repin - Pin uporabnika, ki ga v svoj profil doda en ali več drugih uporabnikov

Povzetek

ETL proces predstavlja širok pojem pridobivanja, preoblikovanja in nalaganja podatkov. Vsaka izmed faz potrebuje podrobno definiran postopek, ki bo podatke prenesel na drugo lokacijo ali preoblikoval v potrebno obliko. Nestrukturirana oblika podatkov in njihova velika količina, ki sta pogosti danes, proces še dodatno otežujeta, kar podaljšuje njegovo izvedbo. S primernim ETL orodjem lahko poenostavimo implementacijo procesa in zagotovimo boljši nadzor nad izvajanjem. V diplomski nalogi se je s takim orodjem hotelo pokazati, kako to poenostavitev narediti v praksi. Primerjani sta dve komercialni in odprtokodni orodji. Izbrano je bilo orodje Talend in nato podrobneje predstavljeno njeno delovanje. Opisano je reševanje problemov z obvladovanjem ter integracijo podatkov pridobljenih s spletnim luščenjem in iz socialnega omrežja Twitter. Za orodje je na koncu opravljena še SWOT analiza.

Ključne besede: opravilo, proces, integracija podatkov, Talend, orodje, podatkovno skladišče.

Abstract

The ETL process presents a broad concept of extracting, transforming and loading data. Each of these phases needs to be well defined to transfer the data efficiently to a different location or transform it into the demanded form. Unstructured forms of data along with its huge volume, which is common nowadays, makes this process even more difficult, and is reflected in the longer execution time. With a suitable ETL tool it is possible to simplify the implementation process and assure better control over it. The thesis describes how to complete such simplifications using an appropriate tool in practice. Two commercial and open source tools were compared. Talend tool was chosen and its workflow was later presented in detail. Handling management and integration problems of data is described, where the used data came from web scraping and the Twitter social network. At the end, a SWOT analysis was made for Talend tool.

Keywords: job, process, data integration, Talend, tool, data warehouse.

Poglavje 1

Uvod

Problematika obvladovanja podatkov je v informacijskem svetu prisotna že od nastanka prvih podatkovnih baz in skladišč. Vsi podatki imajo vir, kjer jih lahko najdemo, cilj in njihovo uporabo pa moramo navadno določiti ali najti sami. Na tej poti jih je navadno potrebno tudi prenesti iz enega ali več virov, jih preoblikovati v skladu s poslovnimi potrebami, da bodo na končni točki uporabni za namen, ki smo si ga izbrali. Celoten proces nemalokrat vsebuje prepreke, s katerimi se moramo soočiti, da podatke pripeljemo na cilj v želeni oziroma zahtevani obliki.

V današnjih časih je ETL proces vse prej kot enostaven prenos podatkov iz ene podatkovne baze v drugo. Med viri podatkov obstajajo velike razlike, tako po zanesljivosti kot tudi po dostopnosti. Poleg tega se bistvene razlike prisotne tudi v strukturi oziroma nestrukturi, količini in vsebini podatkov. Če k naštetim dodamo še spreminjanje v realnem času, lahko kmalu ugotovimo, da je ETL proces v takih okoljih zahteven in kompleksen. Za obvladovanje in integracijo podatkov tako potrebujemo orodje, s katerim bi te procese lahko načrtovali in vodili. Prva taka orodja so se pojavila skupaj z ETL procesi. Ta so bila namenjena le za večja podjetja in kot del celotnega poslovnega sistema. Sčasoma so se razširila tudi na manjša in srednja podjetja, kjer podatki predstavljajo bistveni del posla oziroma so ključni za delovanje drugih delov poslovnega sistema.

S porastom socialnih in senzorskih omrežij ter drugih virov velike količine podatkov so se pojavile tudi nove poslovne priložnosti, kako z analiziranjem, preoblikovanjem in združitvami teh podatkov dobiti nove, zanimive podatke. Medtem ko analitična orodja že predvidevajo določeno urejenost podatkov, pa se podatki, preneseni iz virov, le redko nahajajo v uporabni obliki. ETL proces ima tu pomembno vlogo, saj predstavlja prenos in integracijo podatkov iz vhodnih virov v podatkovno skladišče. Od tam pa so že dostopni za analitična orodja, ki opravijo nadaljnje korake do končnega rezultata.

Prav korak med pridobivanjem in analiziranjem podatkov hočemo osvetliti in ga bomo obravnavali v tej diplomski nalogi. Začeli bomo s seznanitvijo lastnosti ETL procesa v sodobnih okoljih in z njegovimi tipičnimi problemi, s katerimi se srečujemo danes. Izmed številnih orodij, ki se uporabljajo za integracijo podatkov, bomo predstavili nekatere najbolj znane ter izpostavili platformo Talend. Njej se bomo podrobneje posvetili in raziskali možnosti, kje lahko z njeno uporabo poenostavimo ETL proces. Na praktičnih primerih bomo prikazali način reševanja klasičnih problemov ter implementacijo v Talendu. Rezultate bomo sklenili z SWOT analizo in izpostavili druge alternative reševanja uporabljenih primerov.

V nadaljevanju v drugem poglavju najprej sledi predstavitev ETL problematike, današnjih trendov, oblike podatkov, sistemske infrastrukture in nekatera nabolj razširjena ETL orodja. Tretje poglavje je namenjeno platformi Talend, kjer spoznamo njene produkte in delovno okolje. Uporaba Talenda na klasičnih primerih obvladovanja podatkov sledi v četrtem poglavju. Zadnje, peto poglavje vsebuje zaključek, v katerem strnemo rezultate in ugotovitve diplomskega dela.

Poglavje 2

Podatkovno skladišče in ETL proces

Proces ekstrahiranja podatkov iz vhodnih sistemov in njihov prenos v podatkovno skladišče imenujemo ETL, ki označuje pridobivanje (*extraction*), preoblikovanje (*transformation*) in nalaganje (*loading*). Kratica ETL je morda precej preprosta, saj izpušča transportno fazo in daje vtis, da se vsaka od preostalih faz razlikuje. Z omenjanjem ETL upoštevamo tudi zajemaje podatkov. Zavedati se moramo, da se ETL nanaša na širok proces in ne le na tri dobro definirane korake [17].

2.1 Pridobivanje (*Extract*)

Pridobivanje podatkov iz različnih sistemov in virov na pravilen način je pogosto največji izziv, s katerim se soočimo v celem ETL procesu. Od tega dela procesa je precej odvisno, kako uspešna bo naslednja faza - preoblikovanje. V splošnem želimo pridobljene podatke spraviti v eno obliko. Običajni viri podatkov, kot so relacijske podatkovne baze in ploske datoteke (*flat files*), nimajo vedno iste strukture, poleg tega pa podatki prihajajo tudi iz nerelacijskih podatkovnih baz in drugih zunanjih virov, kot so podatki pridobljeni s spletnim luščenjem. Raznovrstnost virov vpliva na kompleksnost samih

podatkov, kar povzroči, da je preoblikovanje potrebno izvesti tudi večkrat zapored [7].

2.2 Preoblikovanje (*Transformation*)

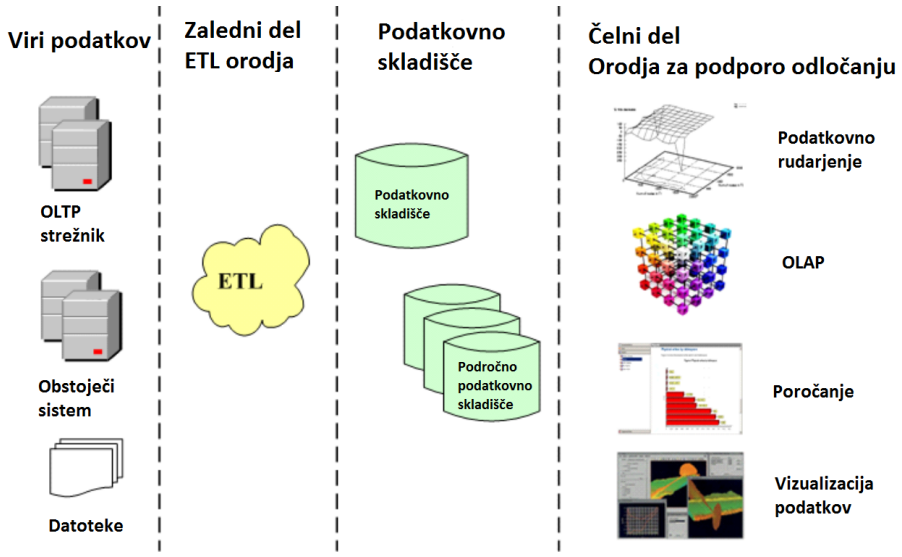
Faza preoblikovanja vsebuje različna poslovna pravila, katera določajo, kako je potrebno podatke preoblikovati, preden jih shranimo na ciljno mesto. Pravila najpogosteje vsebujejo enega ali več od naslednjih preoblikovalnih tipov: izbira samo določenih stolpcev, urejanje (naraščajoče, padajoče), agregacijo, izračun nove vrednosti iz več različnih, razdelitev stolpca na več stolpcev, združevanje različnih virov (*lookup* tabela) ipd [7].

2.3 Nalaganje (*Load*)

Nalaganje podatkov se običajno izvede v podatkovno skladišče, odvisno od poslovnih zahtev. Postopek lahko prepiše prejšnje podatke ali pa opravi posodobitev obstoječih podatkov. Nalaganje se pogosto izvaja na določene časovne intervale, ki so različno dolgi, odvisno od izbrane strategije in časa, ki je na voljo. Pred nalaganjem se po navadi podatke začasno shrani v podatkovno bazo, od kjer se najprej naložijo v operacijsko podatkovno bazo, nato pa še v bazo v podatkovnem skladišču. Tam so podatki razdeljeni naprej v dimenzije in dejstva (*facts*) [7].

2.4 Podatkovno skladišče

Podatkovno skladišče (*data warehouse*) predstavljajo različne tehnologije, s ciljem omogočiti čim boljše in čim hitrejše sprejemanje odločitev osebi, kateri ta naloga pripada. Tipično je razdeljena na čelni del (*front-end*) in zaledni del (*back-stage*). Do prvega dostopa končni uporabnik za uporabo podatkov za podporo odločanju, zaledni del pa je namenjen procesom nalaganja,



Slika 2.1: Abstrakten prikaz arhitekture podatkovnega skladišča

čiščenja in prenosa podatkov, ki poskrbijo za vnos podatkov v podatkovno skladišče [5].

Osnovni cilji podatkovnega skladišča so [27]:

- zagotoviti dostopnost podatkov organizacije,
- zagotoviti konsistentnost podatkov,
- prilagodljivost in prožnost,
- predstavlja varno zgradbo, ki varuje naše informacije in
- predstavlja temelj za sprejemanje odločitev.

V zadnjih nekaj letih je hitro narasla potreba po uporabi podatkov iz podatkovnih skladišč v realnem času. Današnja orodja v poslovni inteligenci in napovedni analitiki potrebujejo sveže podatke za kakovostne rezultate. Pri tradicionalnem pristopu najbolj sveži podatki niso na voljo. S tem se je pojavila tudi potreba, da se posodobitev podatkov ne izvaja več periodično, temveč neprekinjeno. Prav tako je vse večja potreba po ETL orodjih, ki bodo

nudila integracijo podatkov v podatkovna skladišča v realnem času [7]. Ob hitrem naraščanju količin podatkov in krajšanju časa po zahtevi dostopa do podatkov se kot rešitev omenja vpeljava paralelizacije v ETL proces [21].

2.5 Razširjena ETL orodja

Prvotni razlog za razvoj ETL orodij je poenostavitev implementacije in vzdrževanja ETL procesov, kar nam najbolj prizanese pri programiranju. Platforma za nadzor ETL procesa naj bi praviloma obsegala vsa področja zahtev uporabe znotraj organizacije [13].

ETL orodja lahko razvrstimo v dve kategoriji [1]:

- ročno implentirani ETL proces (*hand-coded ETL Process*): Interna ETL orodja, običajno izdelana znotraj podjetja, pogosto v obliki programskih skript, ki vsebujejo zaporedja poizvedb nad podatkovno bazo in druge podprograme. Problem se pogosto pojavi pri enotnosti uporabe, saj lahko v nekem podjetju različni oddelki uporabljajo lastno definiran ETL proces. Ročno implementirani ETL proces je počasnejši in zahteva redne posodobitve [13, 1].

- ETL proces temelječ na orodju (*Tool-Based ETL*): Največkrat vsebuje zmogljiv uporabniški vmesnik, enostaven za uporabo, ki se ga ni posebno težko priučiti. Odpravlja večino težav ročno implemetiranih skript in ponuja komponente za preoblikovanje podatkov, pridobivanje in nalaganje podatkov iz različnih virov ter delovanje na različnih operacijskih sistemih. Današnja ETL orodja pogosto vsebujejo tudi sorodna orodja za podporo poslovni inteligenci ali pa so vsebovana v programski opremi podatkovne baze [1].

Na trgu danes najdemo pester nabor ETL orodij različnih ponudnikov, tako komercialnih kot odprtokodnih [8]. V tej diplomski nalogi bomo primerjali štiri izmed njih, po dva komercialna in odprtokodna glede na izbran kriterij. Primerjavo prikazuje tabela 2.1.

Kriterij	IBM Information Server	Informatica Power Center	Talend Open Studio	Pentaho ETL
Na voljo od	1996	1996	2007	2006
Samostojno ali integrirano	Samostojno	Samostojno	Samostojno	Samostojno
Št. podprtih OS	6	5	7	4
Verzija (l. 2013)	8.1	9.5	5.2	3.2
Okolje ali generirana koda	Oboje	Okolje	generirana koda	generirana koda
Programje kot storitev	Da	Da	Ne	Ne samo-stojno
Enostavnost uporabe	visoka v logičnem smislu	Da	Da	Ne
Ponovna uporabnost	Da	Da	Da	Da
Razhroščevalnik	Da	Da	Da	Ne
Samopopravki sintakse	Da	Delno	Da	Da
Prevajalnik	Da	Delno	Da	Da
Razdeljeni moduli	Ne	Da	Da	Ne
Podatkovni mehanizmi	Beleženje + prožilci	Beleženje	Sporočila + prožilci	Ne
Združevanje tabel	Da	Ne	Da	Ne
Pivotiranje podatkov	Vsa	Vsa	Vsa	Vsa
Vdelani priključki	41	50	35	20

Kriterij	IBM Information Server	Informatica Power Center	Talend Open Studio	Pentaho ETL
Povezave v realnem času	2	6	3	3
Razvrščevalnik opravil	Da	Da	Da	Da

Tabela 2.1: Primerjava štirih razširjenih ETL orodij na različnih kriterijih[1].

- **IBM Information Server** [11] : Osrednji produkt predstavlja IBM InfoSphere Datastage, ki nudi obsežen nabor funkcionalnosti za integracijo podatkov. Z razširitvijo uporabe platforme za različne vrste podatkov, vključno s podatki v velikih količinah in podatkovnimi tokovi, se redno prilagaja spreminjajočim potrebam trga. Skupaj z orodjem Informatica Power Center sta oba na voljo kot storitev v oblaku. Ob današnjem trendu računalništva v oblaku je to pomemebna prednost pred ostalima dvema orodjema, ki tega ne ponujata.

- **Informatica Power Center** [12]: Skupaj z omenjenim IBM produktom, se je na trgu pojavila med prvimi ETL orodji. Gre za eno najbolj prepoznavnih in uporabljenih, saj jo sestavljajo funkcije tako za integracijo podatkov, kakovost in tudi analitiko. Poleg tega je med prvimi omogočala postavitev v oblaku, podporo velikim količinam podatkov in podporo *Map/Reduce* opravilom v Apache Hadoop gruči. Vmesnik omogoča razvijalcem uporabo naprednih integracijskih možnosti, hkrati pa je z prednastavljenimi komponentami za obdelavo podatkov dovolj enostaven tudi za analitike. To omogoča predvsem boljše sodelovanje in izmenjavo podatkov med različnimi uporabniki. Tudi po kriteriju, navedenem v tabeli 2.1, to orodje prevladuje po lastnostih. Kot slabost pa je potrebno izpostaviti, da grafični vmesnik ne omogoča združevanja dveh tabel (*joins*).

- **Pentaho ETL** [18]: Odprtokodni projekt Kettle oziroma Pentaho Data Integration je na tržišču od leta 2006, vgrajen v platformo Pentaho BI Suite za poslovno inteligenco. Tako kot ostala tri orodja se uporablja kot samostojna enota. Poleg odprtokodne je na voljo tudi poslovna različica, pri kateri je s plačilom naročnine zagotovljena dodatna podpora in storitve. Uporablja samostojni javanski pogon, ki opravlja opravila za prenos podatkov iz podatkovnih baz in datotek. V primerjavi z ostalimi obravnavanimi orodji se Pentaho izkaže za nekoliko skromnejše, predvsem pri kriterijih razhroščevalnika, podatkovnih mehanizmih in enostavnosti uporabe. Ker težje konkurira orodjem, ki ju ponujata IBM in Informatica, pa je z usmeritvijo na manjša in srednja podjetja še vedno dovolj konkurenčno.

- **Talend Open Studio** [23]: Vsebuje širok nabor funkcij za obdelavo in prenos podatkov ter učinkovit grafični uporabniški vmesnik. Načrtovanje procesov sloni na uporabi metapodatkov. Ti se shranjujejo v centralizirani shrambi, kar zvišuje skladnost ob načrtovanju novih procesov. Ta je zgrajen na podlagi znanega razvijalskega vmesnika Eclipse. Pri izdelavi opravila se generira programska koda v jeziku Java ali Perl. Podobno kot pri orodju Pentaho, tudi Talend nudi poslovno verzijo, kjer se zaračunava dodatna podpora. Kljub odprtokodnosti konkurira obema komercialnima orodjema. Talend je od ustanovitve leta 2005 tudi sam pridobil široko bazo uporabnikov [1, 8]. Spletna skupnost uporabnikov je odzivna ter predstavlja dober in zanesljiv vir podpore.

  Poleg opisanih štirih orodij na trgu obstajajo še druga, tako odprtokodna kot komercialna orodja. Opis vseh presega obseg te diplomske naloge, bralec pa jih lahko zlahka poišče na spletu. Kot že omenjeno, sta IBM in Informatica s svojima produktoma vodilna na trgu ETL orodij. Zgodnji vstop na tržišče je obema omogočil, da danes njihove produkte uporabljajo največja podjetja in jima s tem dajeta očitno prednost pred konkurenco. Tudi kriterij, ki smo ga uporabili za pri-

merjavo kaže, da sta popolnejša od preostalih orodij. Čeprav je v tem primeru razkorak med obema skupinama viden, nekatere druge primerjave kažejo, da razlika predvsem v funkcionalnosti ni tako očitna [15]. Z novejšimi verzijami se odprtokodna ETL orodja približujejo komercialnim, v prid pa se šteje tudi njihova cenovna ugodnost. Manjša in srednja podjetja lahko z njimi zadostijo svojim visokim ciljem. Izmed naštetih orodij bi radi izbrali enega, s katerim bomo v nadaljevanju lotili reševanja nekaterih klasičnih problemov v ETL procesih. Tudi obe komercialni orodji ponujata brezplačno različico, a se hočemo pri našem delu izogniti omejeni uporabi preizkusnih verzij. Pri primerjavi obeh odprtokodnih orodij se Talend izkaže z boljšo razhroščevalno enoto in detekcijami sintaktičnih napak. Omenjena dva kriterija sta pomembna faktorja, saj sta na primer pri preoblikovanju podatkov in pretvarjanju različnih podatkovnih tipov v veliko pomoč. Pri uporabi ETL orodja si želimo tudi primernega testnega okolja, kjer bomo lahko izkoriščali priključke z drugo programsko opremo in z njimi povezane funkcionalnosti. Talend ponuja testno navidezno okolje, ki vsebuje že prednameščeno ETL orodje in programski paket enega izmed ponudnikov Hadoop distribucij. Naštete lastnosti nam dajejo zadosten razlog, da izberemo Talend. V naslednjem poglavju ga bomo spoznali še podrobneje.

Poglavje 3

Talend

Pod imenom Talend [23] se nahaja več orodij za integracijo (velikih) podatkov, njihovo obvladovanje, kvaliteto in podporo sorodnim storitvam. Talend je prvotno ime podjetja, ustanovljenega leta 2005, ki ponuja omenjeno programsko opremo. Kot glavno značilnost v primerjavi z ostalimi ponudniki na trgu se izpostavlja odprtokodnost produktov. To daje uporabnikom svobodo pri načinu uporabe Talend produktov z namenom, da se čim bolj prilagodijo uporabniškim zahtevam [22]. Kljub odprtokodnosti ga uporabljajo tudi bolj znana podjetja [24].

Z novimi potrebami v industriji pri obvladovanju velike količine podatkov, se je Talend skozi leta razširil v več kot le klasično ETL orodje. V novejših verzijah produktov se nahaja vedno več komponent, ki omogočajo povezavo in uporabo različnih zunanjih virov in storitev. Uporaba Talend produktov je tako v splošnem brezplačna, zaračunava pa se dodatna podpora, kot je svetovanje na mentorski ravni in vodenje večjih projektov integracij podatkov.

3.1 Produkti

Talend svoje produkte razdeli v 4 skupine, pri katerih je vsaka od namenjena določeni obravnavi podatkov:

- **Integracija podatkov (*Data Integration*)**

  Temelji na grafičnem razvojnem okolju Talend Open Studio for Data Integration, ki je derivat znanega razvojnega okolja Eclipse [6]. Z uporabo razpoložljivih komponent zasnujemo opravila (*jobs*), ki sestavljajo ETL proces. Zadnja izdana verzija tega produkta vsebuje preko 800 komponent in priključkov, s katerimi lahko podrobno načrtujemo vsak del ETL procesa [25].

- **Integracija velikih podatkov (*Big data Integration*)**

  Obvladovanje in integracija velikih podatkov tvori v Talendu posebno vejo. Ob že omenjenih funkcionalnostih pri platformi za integracijo "navadnih"podatkov ta verzija vsebuje še nabor komponent za poenostavitev dela z veliko količino podatkov. Omogoča poenostavitev dela s Hadoop distribucijami. Ta produkt, natančneje Talend Open Studio for Big Data, bo v nadaljevanju podrobneje obravnavan, saj bomo v naslednjem poglavju v njem reševali klasične primere obvladovanja velike količine podatkov.

- **Integracija aplikacij (*Application Integration*)**

  Orodje za komunikacijo z drugimi deli distribuiranega poslovnega sistema.

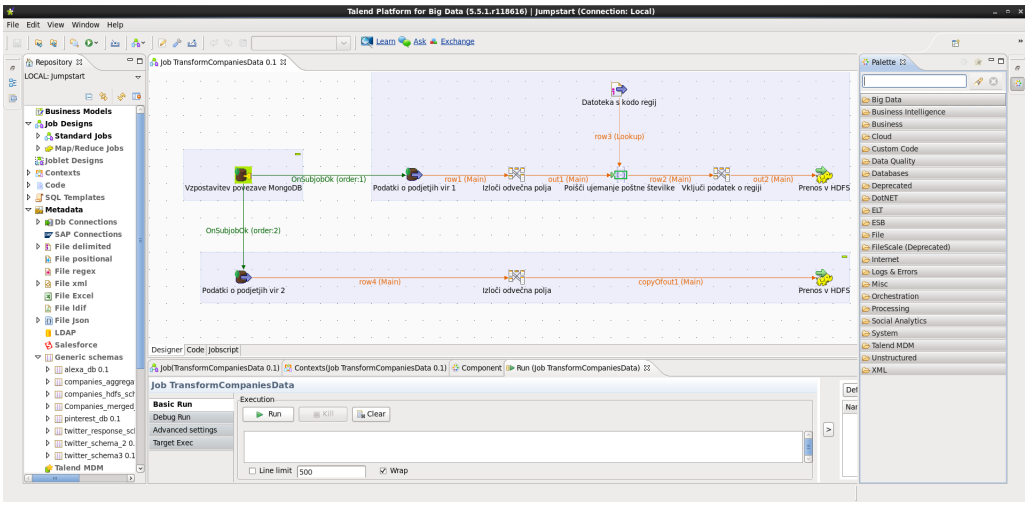
- **Obvladovanje kvalitete podatkov (*Master Data Management*)**

  Uporaba pri zagotavljanju visoke kvalitete oziroma nadzora čistoče poslovnih podatkov. S svojim naborom komponent omogoča dobro izmenjavo podatkov z drugimi uporabniki.

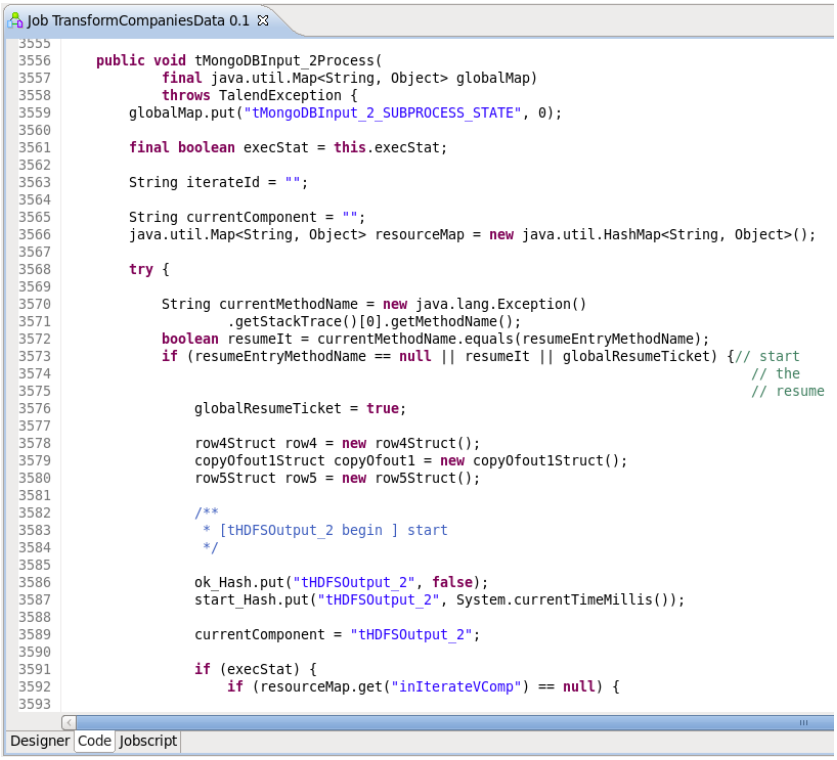
Pri vseh štirih vrstah produktov je na voljo brezplačna, odprtokodna verzija Open Studio, izdana pod licenco Apache Licence v2. Ta v večini ponuja pretežni del vseh funkcij in omogoča, da se uporabnik in hitro enostavno seznani z orodjem ter nadaljnjo uporabo. Talend ponuja brezplačno podporo z uporabo spletne skupnosti, dodatna razširitev in individualna podpora pa sta plačljiva.

3.2 Delovno okolje

Talend Open Studio okolje je zgrajeno na osnovi razvijalskega okolja Eclipse. Za opravila, ki jih uporabniki načrtujejo, se v ozadju generira programska koda v programskem jeziku Java (Slika 3.2). Ta vsebuje vse potrebne podprograme za izvedbo prenosov podatkov. Grafično okolje ponuja že prej omenjen širok nabor komponent in orodij, s katerimi lahko povežemo skoraj vsak vir podatkov. Pri povezovanju komponent si pomagamo z njihovimi nastavitvami. Pri tem nimamo opravka z dejanskimi podatki, ki bodo z opravilom obdelani ali preneseni. Vse nastavitve komponent in celotnega opravila so shranjene v metapodatkovni shrambi. Vsaka komponeneta vsebuje podatkovni shemi za podatke na vhodu in izhodu komponente. Ti dve se lahko razlikujeta, če komponenta opravlja preoblikovanje podatkov. Definiranje sheme je lahko zamudno opravilo, če so podatki predstavljeni z velikim številom stolpcev, kar je pogost primer. Metapodatkovna shramba nam tukaj omogoča uvoz in shranitev podatkovne sheme iz zunanje `.xml` ali `.json` datoteke. Od tam jo lahko uporabimo pri katerikoli komponenti. Pri povezovanju komponent, ki uporabljajo enako podatkovno shemo, definiranje ni potrebno, saj jo lahko naslednja komponenta podeduje od predhodnice. Slika 3.1 prikazuje izgled grafičnega vmesnika med načrtovanjem opravila. Na levi strani se nahaja metapodatkovna shramba, ki vsebuje sezname opravil, uporabniško definiranih povezav na zunanje vire podatkov in vse preostale nastavitve. Sredinski del vmesnika predstavlja plošča za načrtovanje opravila, kjer razporejamo in med seboj povezujemo komponente. Nastavitvam posamezne komponente je namenjen razdelek pod osrednjo ploščo. Na desnem delu se nahaja nabor komponent in orodij, razporejenih po kategorijah.



Slika 3.1: Grafično razvojno okolje Talend for Big Data



Slika 3.2: Primer generirane javanske programske kode opravila v Talendu

Poglavje 4

Reševanje klasičnih problemov obvladovanja podatkov s platformo Talend

4.1 Uporabljena programska oprema

Preden se lotimo reševanja problemov, je najprej vredno predstaviti delovno okolje in uporabljeno programsko opremo. Delovno okolje bo predstavljalo navidezno okolje Talend Big Data Sandbox [19], ki vsebuje že prednameščeno programsko opremo in omogoča takojšno uporabo Talend platforme za velike podatke. Nameščena je tudi gruča Hadoop ponudnika Cloudera [4] z enim podatkovnim vozliščem. Poleg navideznega okolja bomo uporabili še NoSQL podatkovno bazo MongoDB [14], ki je nameščena na gostiteljskem operacijskem sistemu. Shema našega testnega okolja je prikazana na sliki 4.1.

Čeprav mnoge predvsem zanima vsebina podatkov, katero nato na različne načine analiziramo, je pot do nje prav tako pomembna. Z orodjem kot je Talend, želimo korake na tej poti skrajšati in poenostaviti.

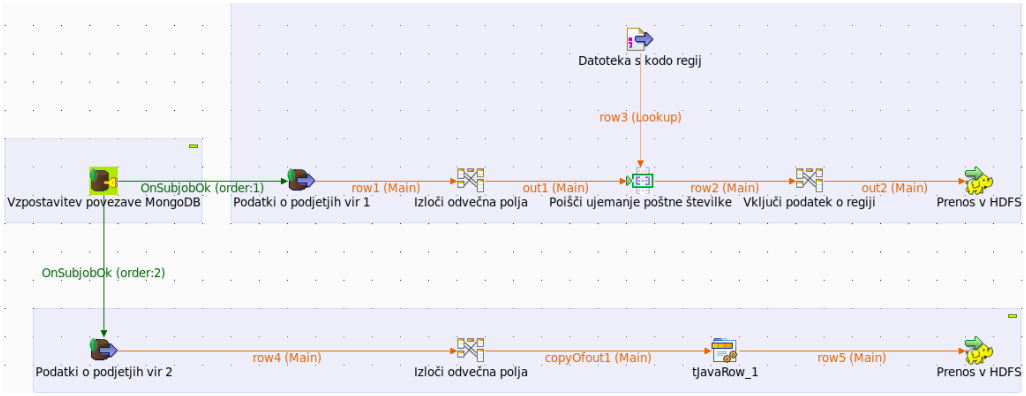
Obravnavali bomo probleme, na katere pogosto naletimo pri preoblikovanju podatkov in njihovem združevanju. S takimi se danes redno srečujemo pri

Slika 4.1: Shema testnega okolja

podatkih, ki izvirajo s spleta in družabnih omrežij. V prvem in tretjem primeru se soočimo z izluščenimi podatki s spleta. V vsakem od njih združevanje predstavlja določen problem, poleg tega pa moramo podatke tekom celotnega procesa večkrat preoblikovati. Drugi primer prikazuje zajemanje podatkov na družabnem omrežju Twitter. Pripraviti jih moramo za možnost takojšnje uporabe, hkrati pa nalagati v podatkovno skladišče.

4.2 Integracija in skladiščenje izluščenih podatkov o podjetjih

Podatki, pridobljeni z luščenjem spletnih strani, imajo pogosto lastnosti velikih količin podatkov, kot so delna strukturiranost, količina in vprašljiva zanesljivost. To so tudi razlogi, ki predstavljajo izziv pri njihovi uporabi in združevanjem z drugimi viri podatkov. V našem primeru bomo uporabili izluščene podatke o podjetjih, ki jih je spletni luščilec uspel pridobiti in shraniti v podatkovno bazo. Z združitvijo podatkov želimo zgraditi svojo zbirko podatkov o podjetjih. Nad njo bomo nato opravili agregacijo in izvozili podatke v datoteko.

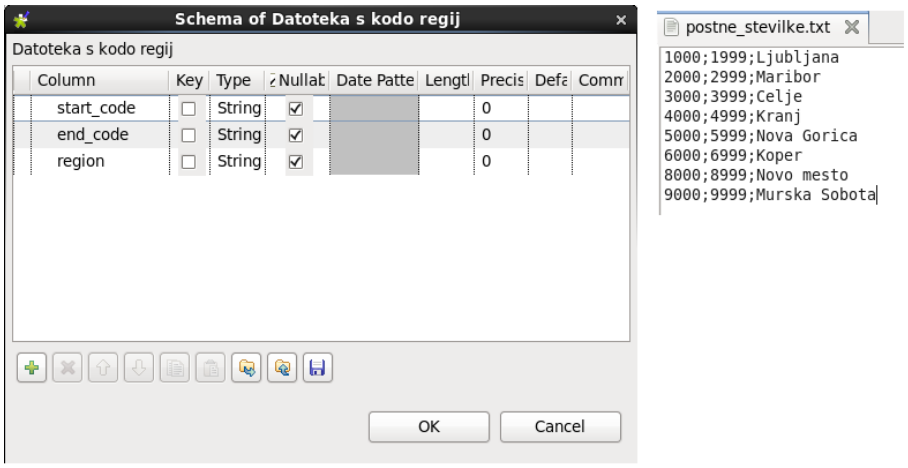


Slika 4.2: Pridobivanje, preoblikovanje in nalaganje podatkov o podjetij

Nalogo bomo razdelili na štiri korake. Za vsakega izmed njih bomo izdelali svoje opravilo. V prvem koraku, prikazanem na sliki 4.2, bomo izluščene podatke prenesli iz podatkovne baze v Hadoop-ov datotečni sistem. Podatki izvirajo iz dveh različnih spletnih virov. Iz prvega vira, ki ga bomo označili kot primarnega, želimo pridobiti osnovne podatke o podjetjih (naziv, naslov, matična in davčna številka, šifra in naziv dejavnosti). Drugi vir je namenjen kontaktnim podatkom podjetij. V komponenti za vzpostavitev povezave s podatkovno bazo MongoDB definiramo povezavo z vnosom naslovnih parametrov in avtentikacije. Oba vira podatkov se nahajata v isti podatkovni bazi, a v ločenih zbirkah (*collections*). Na tem mestu se opravilo razdeli na dve podopravili, kjer pridobivanje podatkov iz vsake izmed zbirk oziroma virov obravnavamo posebej. Vrstni red izvajanja podopravil v tem primeru ni pomemben. Za vsak vir uporabimo komponento `tMongoDBInput`, v katero vnesemo poizvedbo za pridobitev podatkov. Vnesti moramo ime zbirke, poizvedbeni stavek in podatkovno shemo (polja dokumenta). V našem primeru ne operiramo z veliko količino podatkov, zato lahko uporabimo privzeto poizvedbo, ki izbere vse dokumente iz zbirke. Tudi podatkovna shema vsebuje dovolj majhno število polj, da jo lahko definiramo ročno. Shemi, ki ju definiramo, je smiselno shraniti v metapodatkovno shrambo. Ko ju bomo potrebovali drugje, nam ju ne bo potrebno ponovno definirati.

Podatki so zaradi svoje delne strukturiranosti potrebni obdelave pred končno shranitvijo v ciljno mesto. Iz dokumenta izberemo relevantna polja in odstranimo odvečno vsebino, ki je spletni luščilec ni uspel odstraniti sam. Vhodno podatkovno shemo komponente `tMap` lahko iz predhodne podedujemo. Polj, ki jih na tem mestu ne potrebujemo, ne povežemo v izhodno shemo. Prav tako želimo v tem koraku iz naslova (poštne številke) podjetja določiti regijo, iz katere prihaja. Ta bo predstavljena z novim poljem v podatkovni shemi. Za določitev regije iz poštne številke uporabimo komponento `tIntervalMatch` in *lookup* datoteko, kjer je vsaka izmed regij definirana z razponom poštnih številk. V komponenti določimo stolpec, katerega vrednost primerjamo, za datoteko pa stolpca s spodnjo in zgornjo mejo intervala ter stolpec z vrednostjo, ki ga interval predstavlja. Slika 4.3 prikazuje shemo in vsebino *lookup* datoteke. Podjetju določimo regijo na podlagi uvrstitve njegove poštne številke iz naslova. Tudi pri preoblikovanju podatkov iz drugega vira uporabimo komponento `tMap` za izbiro potrebnih stolpcev. Odstranjevanje odvečne vsebine iz vrstice oziroma posameznih stolpcev je v našem primeru precej specifično. V vsakem stolpcu se namreč nahaja drugačna vsebina (niz znakov), ki jo je potrebno izločiti. Odločimo se, da bomo v ta namen uporabili komponento `tJavaRow`, kjer sami definiramo način, kako bo komponenta obdelala vrstice.

Za nalaganje podatkov v datotečni sistem Hadoop uporabimo komponento `tHDFSOutput`. Za nastavitev povezave do Hadoop gruče uporabimo nastavitve iz metapodatkovne shrambe, ki so že vnaprej definirane s strani testnega okolja. Podatkovne sheme ne spreminjamo in jo podedujemo od prejšnje komponente. Vseeno je podatkovno shemo, ki jo bodo imeli podatki v podatkovnem skladišču, dobro shraniti. Nanjo se bomo namreč lahko sklicevali, ko bomo te podatke zopet uporabili. Določiti moramo še, kako bodo naši podatki predstavljeni v podatkovnem skladišču. V našem primeru bomo podatke naložili samo enkrat in bomo zato izbrali tekstovno datoteko. V primeru, da bi opravilo izvajali večkrat zaporedoma ter tako dodajali nove podatke, bi bilo smiselno vsakič ustvariti novo datoteko ali dodati podatke

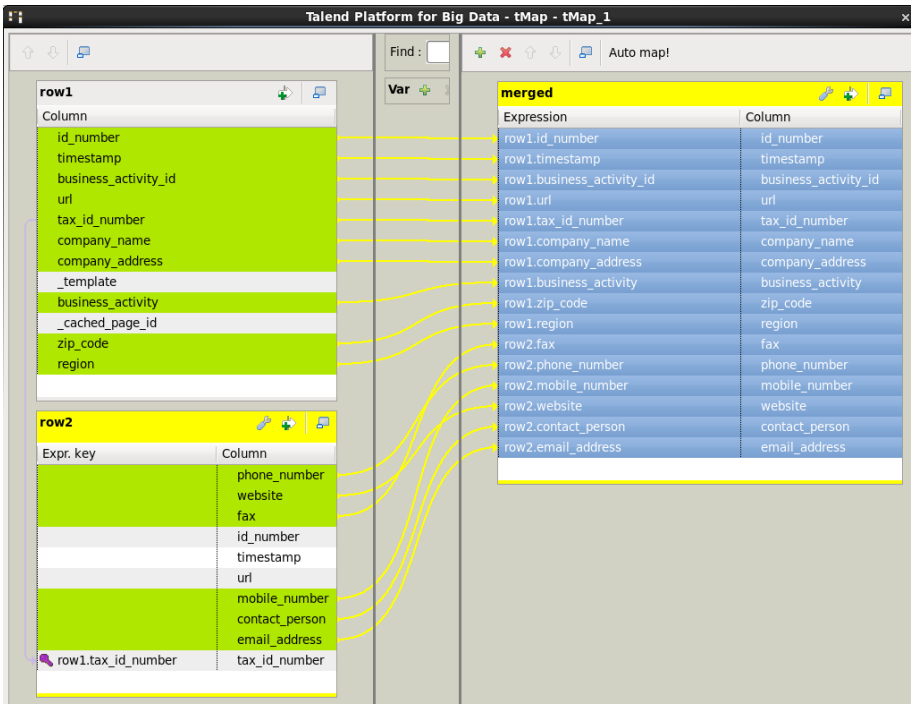


Slika 4.3: Shema in vsebina *lookup* datoteke

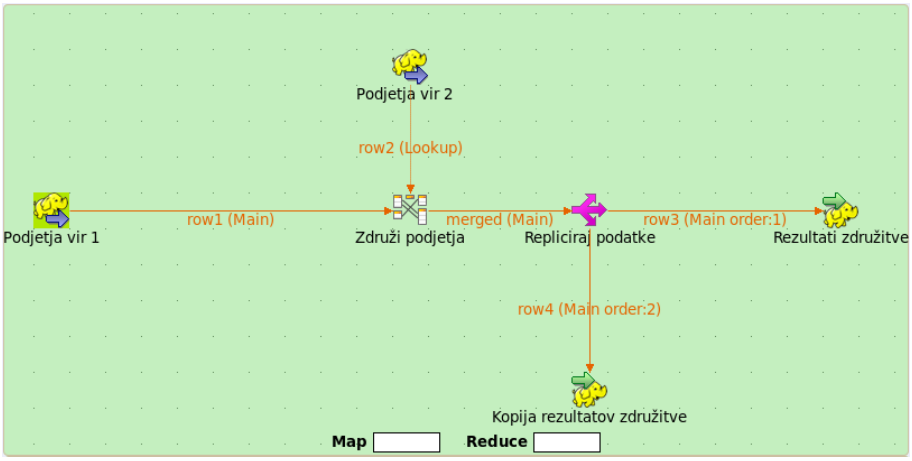
obstoječi.

Za drugi korak izdelamo opravilo, ki bo podatke podjetij iz obeh datotek združil v celoto. S komponentama `tHDFSInput` izberemo datoteki, v kateri smo v prejšnjem opravilu naložili podatke. Poiskati pravilo oziroma ključ, po katerem bodo podatki združeni, je lahko zapleten problem, če se viri vsebinsko precej razlikujejo. V našem primeru imamo to srečo, da v obeh virih najdemo davčno številko podjetja, ki je edinstvena za vsako podjetje in jo tako lahko uporabimo pri ujemanju (Slika 4.4). Opravilo, ki ga izdelamo, je pravzaprav *Map/Reduce* opravilo, ki se bo v celoti izvedlo v Hadoop gruči. Talend loči opravila med standardnimi in *Map/Reduce*, kjer ponuja tudi omejen nabor komponent. V Talendu bomo tako dobili le informacije o poteku izvajanja in izhodni status [26]. Združeni podatki so kreirani v novi datoteki. Za potrebe ohranjanja različnih verzij združevanj datoteki v imenu dodamo časovni žig, ob tem pa ustvarimo še njeno kopijo z uporabo komponente `tReplicate`. Za shranjevanje obeh datotek uporabimo komponento `tHDFSOutput` kot v prejšnjem opravilu. Slika 4.5 prikazuje celotno opravilo.

V naslednjem koraku opravimo agregacijo nad podjetji, ki je namenjena kasnejšemu izvozu podatkov iz Hadoopovega datotečnega sistema. Denimo, da želimo imeti podjetja združena po posameznih regijah, kar nam bo olajšalo



Slika 4.4: Združitev obeh virov po polju davčna številka in določitev nove sheme podatkov



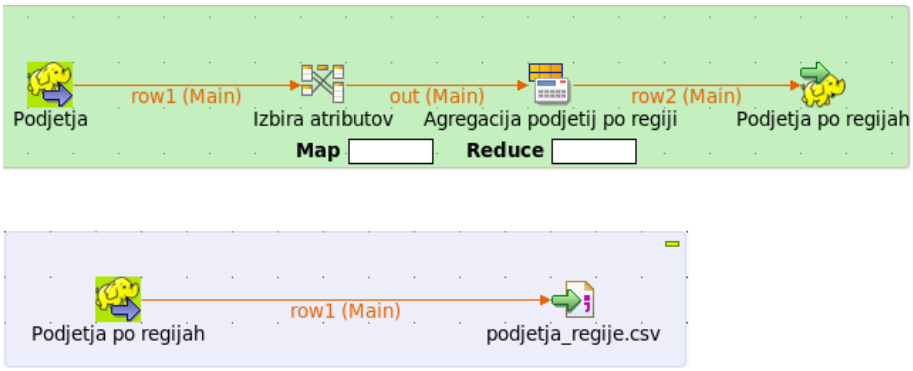
Slika 4.5: Združitev podatkov in shranitev v Hadoop

nekaj dela, ko bomo v prihodnosti opravljali analizo za katero izmed njih. Za to nalogo zopet izdelamo *Map/Reduce* opravilo. Za agregacijo podatkov uporabimo komponento `tAggregateRow`. Izbrati moramo stolpec, po katerem bomo vrstice združevali. V našem primeru bo to stolpec, ki opisuje regijo. Določiti moramo še, kaj storiti z ostalimi stolpci. Komponenta vsebuje različne funkcije, ki jih lahko uporabimo: *count*, *min*, *max*, *avg*, *sum*, *first*, *last*, *list*, *list (objects)*, *count (distinct)* ali *standard deviation*. Odločimo se, da bomo ostale podatke podjetij strnili v seznam (*list*). Preoblikovane podatke shranimo v novo datoteko.

V zadnjem, čertem koraku, ustvarjeno datoteko izvozimo iz Hadoopovega datotečnega sistema v novo `.csv` datoteko. Pri tem uporabimo komponento `tFileOutputDelimited`.

4.2.1 Razhroščevanje opravila in obvladovanje napak

Izdelava opravila je kljub jasnosti vmesnika in enostavnosti povezovanja komponent lahko zahtevna naloga. Zgodi se namreč, da med izvajanjem opravila pride do napak. V primeru, da gre za sintaktično ali vhodno izhodno napako, jo bo razhroščevalnik zaznal, prekinil delovanje in nas o tem obvestil.

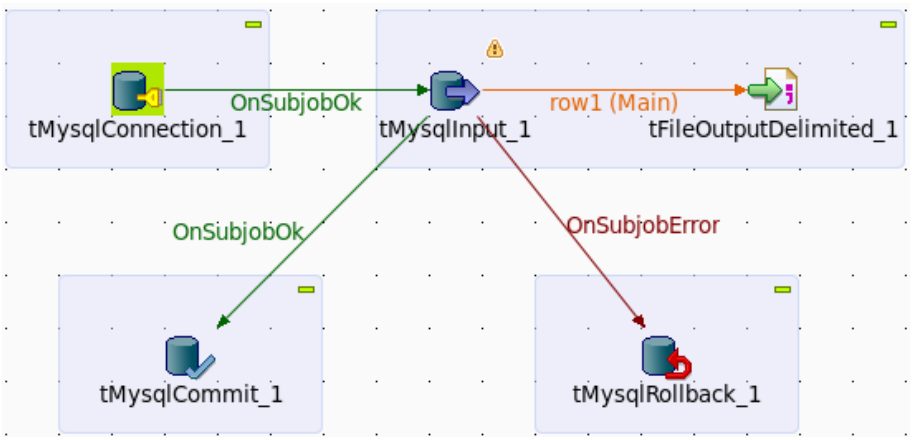


Slika 4.6: Opravili za agregacijo podjetij po regijah in izvoz `.csv` datoteko

Odkrivanje semantičnih napak pa je bolj zapleteno, saj jih bomo opazili le s preverjanjem preoblikovanih podatkov ali šele kasneje pri njihovi analizi. Da se takim problemom izognemo, je pri izdelavi koristno uporabiti vgrajeni razhroščevalnik. Na komponento lahko postavimo zaustavitveno točko (*breakpoint*), kjer bo razhroščevalnik začasno zaustavil izvajanje opravila. Postavimo jo lahko tudi znotraj generirane programske kode, če nas izvajanje še natančneje zanima. Smiselno je zaustaviti opravilo po združevanju več stolpcev v enega, kjer lahko preverimo, ali je združitev dala pričakovano obliko. Razhroščujemo lahko tudi vsako preneseno vrstico posebej, tako da za komponento, ki jo razhroščujemo, povežemo komponento `tLogRow`. Ta bo vsako preneseno vrstico izpisala v terminalno okno.

Talend ne nudi razhroščevanja *Map/Reduce* opravil. Enostavnost razhroščevanja je pomembna lastnost ETL orodja. Z njim si pri načrtovanju lahko prihranimo precej časa, potrebnega za izdelavo opravila. Zmanjšamo tudi možnost, da bi bila končna oblika podatkov po nalaganju napačna.

Pri dostopih do zunanjih virov, kot so datoteke, podatkovne baze ali storitve v oblaku, moramo upoštevati morebitni izpad vira. V primeru, da med prenosom podatkov iz podatkovne baze pride do napake, je potrebno zagotoviti, da v podatkih ne pride do neskladja. Običajen način obvladovanja take napake je razveljavitev transakcije (*rollback*). S tem vrnemo podatkovno bazo v predhodno, stabilno stanje. Talend ima za večino podatkovnih baz,



Slika 4.7: Primer uporabe komponente za razveljavitev prenosa ob napaki.

ki jih podpira, komponento za razveljavitev transakcije. Primer uporabe je prikazan na sliki 4.7. Pri prenosu podatkov iz podatkovne baze v datoteko komponenti `tMySqlInput` določimo prožilec na dogodek, ki se lahko zgodi ob njenem delovanju. V našem primeru sta definirana dva prožilca, kjer vsak od njiju sproži podopravilo. Prvi se nanaša na uspešen prenos podatkov, ki bo izvedel podopravilo `OnSubjobOk` za potrditev prenosa. Ob napaki pri prenosu, pa se bo izvedlo podopravilo `OnSubjobError`, v katerem se nahaja komponenta `tMySqlRollback`. Ta bo preprečila potrditev prenosa, ki ni bil dokončan.

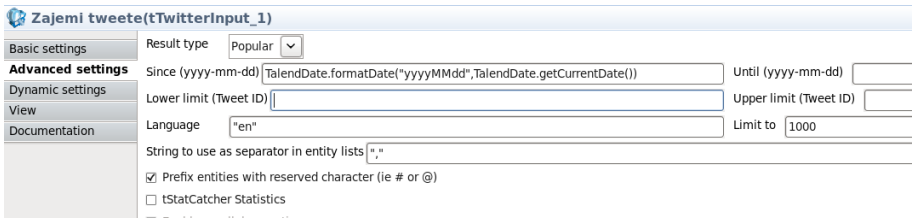
4.3 Zajemanje, obdelava in shranjevanje odzivov s socialnega omrežja Twitter

Denimo, da želimo slediti omembam določenih oseb ali podjetij ter primerjati, katero od njih je ob določenem času bolj popularno, kdo so osebe, ki jih omenjajo in katere vsebine se še pojavljajo ob njihovem imenu. Na dnevni ravni želimo izbrati določeno število najbolj popularnih tweetov in opraviti enostavno primerjavo (popularnost v številu retweetov opazovane ključne besede, pripadajoči hashtagi), za širšo pa tweete tudi shranimo v podatkovno

skladišče Hadoop. Tam bodo na voljo za kasnejše, podrobnejše analize za pretekla obdobja.

Talend ne vsebuje posebne komponente za uporabo Twitter API, zato za zajem tweetov uporabimo neuradno komponento `tTwitterInput` [28]. Ta omogoča dva načina zajemanja: zajem tweetov po določenem kriteriju (najnovejši, popularni) ali priključitev na podatkovni tok, kjer zajemanje traja do določenega števila tweetov. Pri slednjem se nam lahko zgodi, da tweeti z vsebovano ključno besedo niso pogosti, nastavljena meja števila zajetih tweetov pa je visoka. Tako bo zajemanje trajalo zelo dolgo in s tem upočasnilo celotno opravilo. Pri prvem načinu zajemanja pa zajemamo tweete, ki so že bili objavljeni. Ta način bomo uporabili za naš primer. V nastavitve komponente določimo kriterij, katere tweete želimo prejeti. Odločimo se za zajem popularnih tweetov trenutnega datuma, v angleškem jeziku, z zgornjo mejo največ 1000 tweetov za posamezno ključno besedo. Tu naj omenimo, da upoštevamo Twitterjev algoritem določevanja popularnosti tweetov [10]. Opisane nastavitve komponente so prikazane na sliki 4.8. Poleg omenjenih nastavitev določimo še podrobnosti tweeta (npr. uporabnik, sporočilo, vsebovani hashtagi), katere želimo prejeti in ključno besedo, katera mora biti vsebovana. Poleg `tTwitterInput` komponente, moramo za njeno delovanje uporabiti še komponento za overjanje in vzpostavljanje povezave, po opravljenjem zajemu pa še komponento za končanje povezave.

Med zajemanjem tweetov opravilo čaka. Šele ko komponenta konča svoje delo, se opravilo nadaljuje z naslednjim korakom. Za pravkar zajete tweete moramo opraviti naslednji nalogi: nalaganje v podatkovno skladišče in priprava agregiranih podatkov za takojšno primerjavo. S komponento `tMap` razdelimo opravilo na podopravili. Zaporedje izvajanja podopravil ni pomembno, saj sta med seboj neodvisni. Za tweete želimo narediti primerjavo, koliko retweetov ima vsaka ključna beseda in kateri hashtagi se radi ponovijo. Opraviti moramo agregacijo po ključni besedi, sešteti število retweetov vsakega tweeta, hashtage pa strniti v seznam. V komponenti `tAggregateRow` uporabimo funkcijo `sum` za vsoto vseh retweetov in `list` za združitev hash-



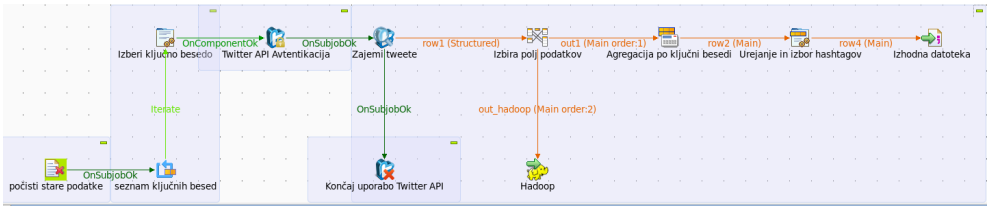
Slika 4.8: Nastavitve komponente Twitter

tagov v seznam. Tukaj podamo še dodaten kriterij, da nas zanimajo samo hashtagi, ki se ponovijo vsaj trikrat. To funkcionalnost realiziramo z lastno programsko kodo, katere vnos omogoča komponenta `tJavaRow`. Rezultate agregacije shranimo v začasno datoteko, ki bo vsebovala vsebino v naslednjem formatu: `ključna_beseda;vsota_vseh_retweetov;hashtagi`.

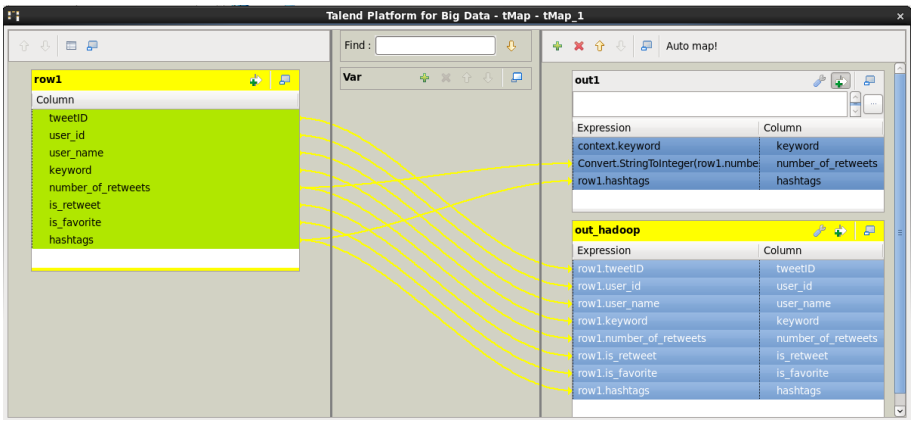
V podatkovno skladišče bomo naložili vse podatke o tweetih, ki smo jih zajeli. Ker bomo tweete zajemali vsakodnevno, je smiselno določiti strukturo imena datotek, ki jih bomo ustvarjali. Uporabimo lahko naslednji format za poimenovanje imen:
`/ključna_beseda/ključna_beseda_časovni_žig`.

Opravilu moramo dodati še funkcionalnost, kako celoten postopek izvršiti za poljubno število ključnih besed. Te bi radi imeli definirane na enem mestu, kjer bi jih lahko enostavno dodajali ali spreminjali. Uporabili bomo komponento `tForEach`, kjer lahko ključne besede dodamo na seznam. Komponenta deluje kot *for each* programska zanka, tako da se bo zaporedje korakov, ki tej komponenti sledijo, ponovilo za vsako ključno besedo (element seznama). V opravilu jo moramo postaviti pred komponento za overjanje Twitter API. Ustvarimo še globalno spremeljivko, ki bo vsebovala trenutno ključno besedo iz seznama. Nanjo se lahko nato sklicujemo na mestih, kjer jo v opravilu potrebujemo. Na sliki 4.9 je prikazano celotno opravilo.

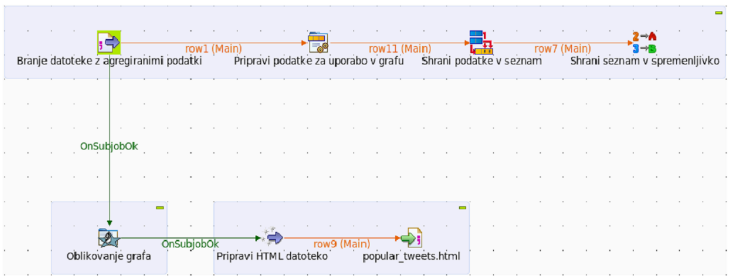
Rezultate naše primerjave bomo podali v HTML datoteki, z uporabo orodja Google Charts. Izdelati moramo še eno opravilo, v katerem preberemo agregirane podatke iz datoteke in ustvarimo novo, HTML datoteko s potrebno programsko kodo za upodobitev grafa. S spletnim brskalnikom

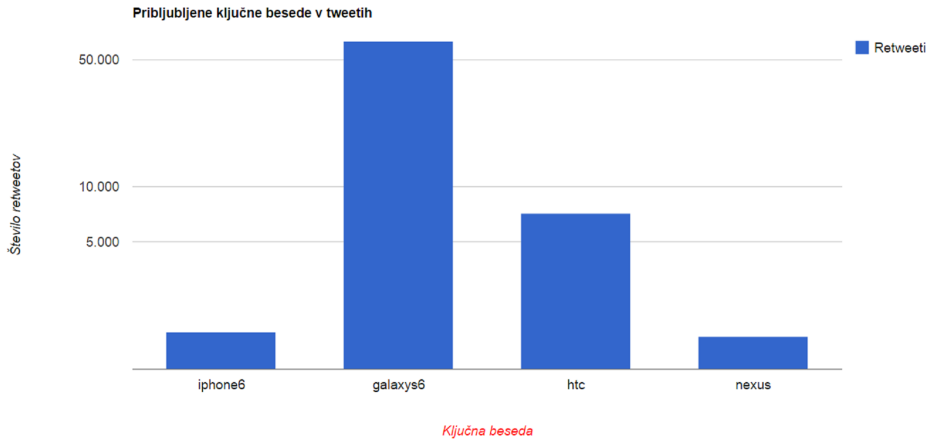


Slika 4.9: Opravilo zajema tweetov



Slika 4.10: Izbira podatkov tweeta za shranjevanje v Hadoop in za primerjanje popularnosti ključnih besed





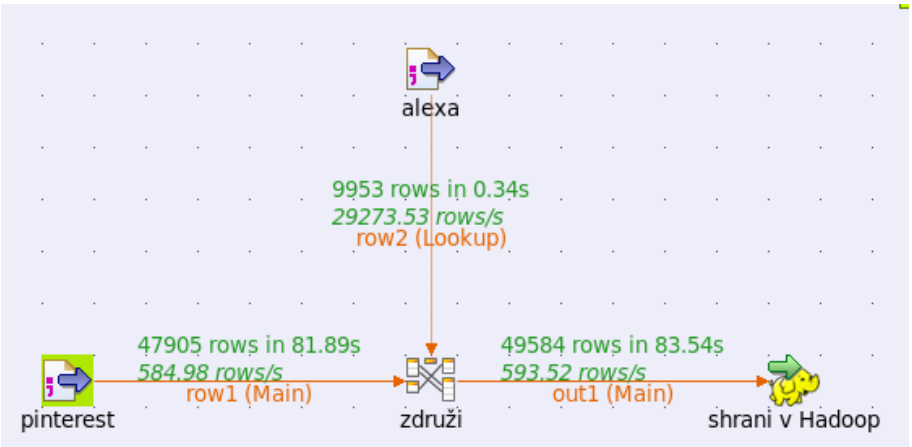
	Ključna beseda	najbolj pogosti hashtagi
1	iphone6	#GalaxyS6 #ArdteistRnaG #iPhone6 #RT #MWC2015
2	galaxys6	#GalaxyS6edge #iPhone6 #Samsung #GalaxyS6 #MWC2015 #NextIsNow #PFW #BBCGoFigure #GalaxyS6Edge #Unpacked
3	htc	#InventForBetter #HTCOneM9 #Valve #MWC15 #HTCOneLife
4	nexus	

Slika 4.11: Opravilo za izdelavo HTML datoteke in graf

lahko nato pogledamo končne rezultate. Opravilo in predstavitev tweetov v grafu sta podana s sliko 4.11.

4.3.1 Izvoz opravila za periodično izvajanje

Za vsak zajem tweetov moramo opravilo posebej izvršiti in prav tako tudi drugo opravilo za grafični prikaz. Ker bi radi shranjevali tweete za neko nadaljnje obdobje, se bosta morali opravili izvršiti periodično (npr. dnevno). Talend platforma v osnovi ne ponuja razvrščevalnika izvajanja (*scheduling*). Uporabimo lahko tistega, ki ga nudi operacijski sistem računalnika, kjer se bo

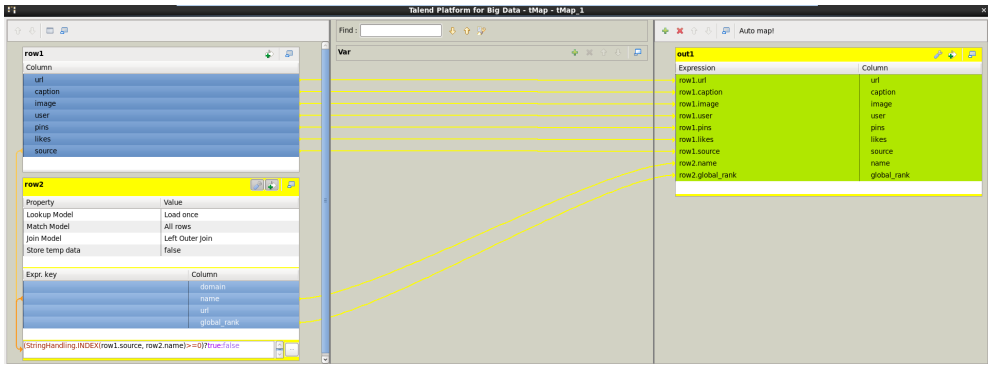


Slika 4.12: Opravilo, ki združi podatke s Pinteresta in Alexe

opravilo izvajalo (npr. na Unix sistemih lahko uporabimo *crontab* in določimo kdaj se bo opravilo izvršilo). Opravilo lahko izvozimo kot samostojno zagonsko skripto, tako za Windows kot tudi Unix okolja. Na operacijskem sistemu mora biti že nameščen Java SDK paket. Da ne izvažamo obeh opravil posebej, ju lahko združimo v eno, glavno opravilo (t.i. *master job*). Z zagonom glavnega opravila se bosta izvršili obe podopravili.

4.4 Primerjava priljubljenosti objav na Pinterestu z obiskanostjo izvirnih spletnih strani vsebine objav

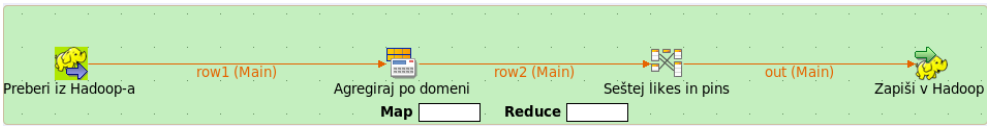
Z razvojem socialnih sistemov ter omrežij je strmo naraslo tudi deljenje medijskih vsebin, kot so slike in videoposnetki [3]. Eden izmed mnogih načinov za objavljanje in deljenje omenjenih vsebin je tudi Pinterest [16]. Na njem lahko uporabnik objavi pin, ki npr. vsebuje zanimivo sliko ali video posnetek, ki ga je našel na neki spletni strani. Hkrati je možno videti tudi pine drugih uporabnikov in jih ponovno pripeti na svoj profil. Vsak pin je možno označiti tud kot priljubljenega.



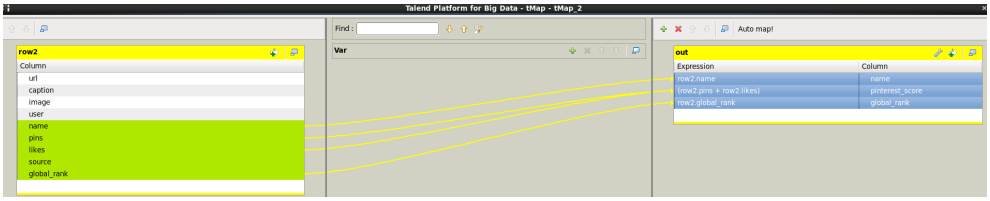
Slika 4.13: Shema združevanja podatkov s pravilom za povezovanje

Radi bi izvedeli, ali se priljubljenost vsebine pina odraža tudi z priljubljenostjo oziroma obiskanostjo spletne strani, s katere vsebina izvira. Za merjenje priljubljenosti na Pinterestu bomo vzeli vsoto repinov in všečkov, za priljubljenost vira pa uvrstitev na lestvici obiskanosti, ki jo najdemo na spletni strani Alexa [2]. Tudi v tem primeru imamo na voljo podatke, pridobljene s spletnim luščenjem. Luščilec je naključno obiskal približno petdeset tisoč različnih pinov ter izluščil naslednje podatke: spletni naslov pina, ime uporabnika, ime objave, vsebino objave (npr. sliko), spletni naslov vira, število repinov in število všečkov. Za vsak spletni naslov vira pa je z Alexe izluščil ime domene in uvrstitev na lestvici obiskanosti. Domene virov se v pinih večkrat ponovijo, tako da je bilo različnih virov približno petkrat manj kot pinov. Podatki, izluščeni iz obeh strani, so ločeni v dveh datotekah, ki ju moramo združiti.

Obe datoteki povežemo v komponento `tMap`, s katero ju bomo združili (Slika 4.12). Datoteka s podatki z Alexe ima vlogo *lookup* tabele, poiskati pa moramo ključ, po katerem bomo podatke združili. Združevanje z relacijo 1:1, kot v primeru s podjetji, bi bilo tukaj napačno, saj si neko domeno lahko lasti več virov pinov. Kot edina možnost se izkaže, da oba vira združimo z ujemanjem imena domene kot podniza v spletnem naslovu vira pina. Shema in pravilo združevanja sta prikazana na sliki 4.13. Vsak pin zdaj vsebuje še podatka o imenu domene in uvrstitvi domene. Preden bomo lahko primerjali



Slika 4.14: Opravilo, v katerem opravimo agregacijo nad imenom domene

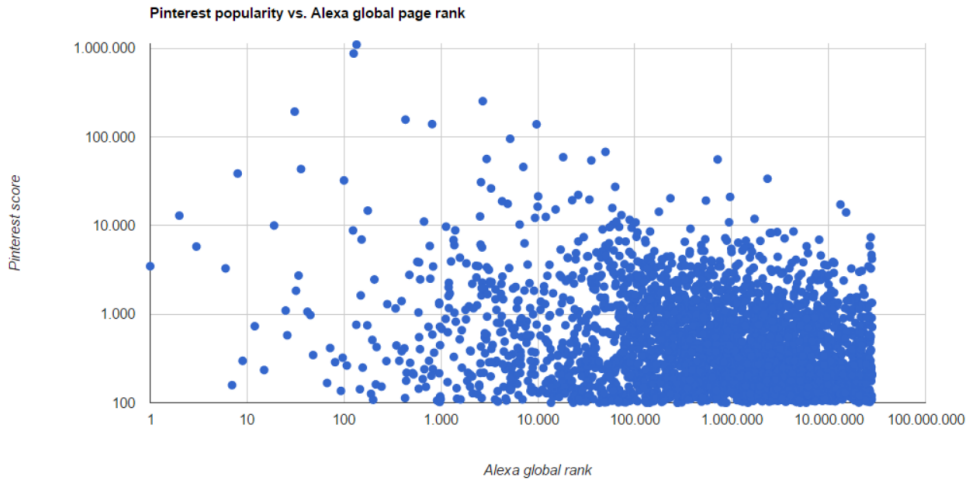


Slika 4.15: Shema združevanja podatkov, ki jih bomo uporabili po agregaciji

priljubljenost in uvrstitev domene, moramo podatke še dodatno preoblikovati. Pred tem jih naložimo v podatkovno skladišče, saj jih bomo uporabili v novem opravilu.

Nad združenimi podatki moramo opraviti agregacijo po imenu domene. Priljubljenost pinov bomo predstavili z enim stolpcem, v katerega bomo sešteli repine in všečke. Poimenovali ga bomo `pinterest_score`. Za nov stolpec in oceno z Alexe (`global_rank`) bomo pri agregaciji uporabili funkcijo `sum`. Agregirane podatke shranimo naložimo v Hadoop. *Map/Reduce* opravilo in podatkovna shema sta prikazani na slikah 4.14 in 4.15.

Podatki sedaj vsebujejo ime domene, priljubljenost na Pinterestu in uvrstitev na lestvici obiskanosti. Omeniti velja, da se meri merita drugače: Nižja vrednost na Alexi pomeni boljšo uvrstitev strani, nižja vrednost na Pinterestu pa slabšo priljubljenost. Grafično bomo primerjavo priljubljenosti predstavili z raztresenim grafom, prikazanim na sliki 4.16. Za izbiro podatkov in pripravo HTML datoteke lahko izhajamo iz opravila, kot je prikazano na sliki 4.11 iz prejšnjega primera. Za optimalno grafično predstavitev bi bilo potrebno podatke dodatno normalizirati, saj je razpon pri uvrstitvi na Alexi precej dolg, vseh točk (domen) pa nekaj manj kot deset tisoč. Prav tako bi z dodatnim urejanjem podatke še natančneje analizirali.



Slika 4.16: Priljubljenost virov vsebin na Pinterestu

4.5 Komponente

Talend vsebuje veliko število komponent in priključkov, ki jih lahko uporabimo pri izvedbi opravil. Razdeljeni so po kategorijah, namenu uporabe in sorodnosti. Nekatere komponente so na voljo samo za uporabo pri standardnih ali *Map/Reduce* opravilih, nekatere pa lahko uporabimo za obe vrsti opravil. V tabeli 4.1 so navedena tista, ki smo jih pri izdelavi opravil najbolj pogosto uporabili.

Komponenta	Kategorija	Standarno in/ali Map/Reduce opravilo
tAggregateRow	Obdelava podatkov	Oboje
tConvertType	Obdelava podatkov	Standardno
tFileDelete	Datoteke	Standardno

 tFileInputDelimited	Datoteke	Standardno
 tFileOutputDelimited	Datoteke	Standardno
 tForEach	Orkestracija	Standardno
 tHDFSInput	Hadoop HDFS	Oboje
 tHDFSOutput	Hadoop HDFS	Oboje
 tIntervalMatch	Kvaliteta podatkov	Standardno
 tJavaRow	Prog. koda po meri	Standardno
 tMap	Obdelava podatkov	Oboje
 tMongoDBConnection	Velike količine podatkov	Standardno
 tMongoDBOutput	Velike količine podatkov	Standardno
 tMongoDBInput	Velike količine podatkov	Standardno
 tReplicate	Orkestracija	Standardno

 tTwitterInput	Analiza socialnih omrežij	Standardno
 tTwitterOAuth	Analiza socialnih omrežij	Standardno
 tTwitterOAuthClose	Analiza socialnih omrežij	Standardno

Tabela 4.1: Tabela komponent

Poglavje 5

Zaključek

5.1 SWOT analiza

SWOT (***S**trenghts, **W**eaknesses, **O**pportunities, **T**hreats*) analiza nam omogoča predstavitev prednosti, slabosti, priložnosti in nevarnosti nekega produkta. Pri tem upoštevamo notranje in zunanje dejavnike, ki vplivajo na produkt. Opravljanje analize nam odkrije stvari, ki se jih je dobro zavedati pred uporabo produkta. Njena dobra izvedba nam lahko pomaga pri strateškem načrtu in sprejemanju odločitev [20]. SWOT analiza za platformo Talend je prikazana na tabeli 5.1 [22, 9].

5.2 Sklepne ugotovitve

V diplomski nalogi smo na obravnavanih primerih prikazali, kako lahko z uporabo platforme Talend poenostavimo pridobivanje, preoblikovanje in nalaganje podatkov v končno podatkovno skladišče. Izbrali smo si različne vire podatkov, kot so spletno luščenje in družabna omrežja, ki imajo vsak svoje svoje značilnosti. S komponentami, ki jih ponuja Talend, smo podatke na različne načine preoblikovali in združili, da so postali razumljivešji in uporabni za nadaljnje operacije nad njimi. Z nalaganjem v podatkovno skladišče Hadoop smo izkoristili Talendove komponente za delo z veliko količino po-

	Pozitivno	Negativno
Notranji dejavniki	Prednosti: 1. široka povezljivost in funkcionalnost 2. strma učna krivulja pri uporabi vmesnika 3. konkurenčna cena glede na druga orodja	Slabosti: 1. brezplačna verzija nima svojega razvrščevalnika opravil 2. enouporabniški način delovanja 3. problematična podpora
Zunanji dejavniki	Priložnosti: 1. boljša prepoznavnost 2. uporaba paralelizma 3. razvoj novih komponent za integracijo	Nevarnosti: 1. skromna podpora pri zagotavljanju kvalitete podatkov 2. počasno prilagajanje trendom za obvladovanje velikih podatkov

Tabela 5.1: SWOT analiza Talenda

datkov. Na koncu smo za Talend naredili še SWOT analizo, s katero smo jasno izpostavili prednosti, slabosti, priložnosti in nevarnosti.

Obravnavani primeri ne sodijo med zelo zahtevne probleme ETL procesa. Količina podatkov s katero smo opravila testirali ni bila velika in tako ni povzročala večjih problemov pri uporabi. Uporaba Hadoop gruče za podatkovno skladišče, ki svoje zmogljivosti pokaže predvsem na velikih količinah podatkov in zahtevnimi *Map/Reduce* nalogami, se morda zdi pretirana. Kljub temu je tudi virtualno okolje z omejenimi sistemskimi viri, kakršno smo uporabljali, dovolj dobro za prikaz možnosti, ki jih Talend ponuja. Nadgradnje diplomskega dela bi lahko šle v smeri uporabe Talenda pri obravnavi hitrih podatkovnih tokov in obdelave v skoraj realnem času. Za našteti dve problematiki bi si zagotovo želeli poenostavitev pri uporabi.

Literatura

[1] A. S. Pall, J.S. Khaira. A comparative Review of Extraction, Transformation and Loading Tools. In: Database Systems Journal vol. IV, št. 2/2013

[2] Alexa. [Online]. Dosegljivo: https:http://www.alexa.com/. [Dostopano 8. 3. 2015].

[3] Oded Nov, Mor Naaman and Chen Ye. Analysis of participation in an online photo-sharing community: A multidimensional perspective. In: Journal of the American Society for Information Science and Technology Volume 61, Issue 3, pages 555–566, March 2010

[4] Cloudera. [Online] Dosegljivo: http://www.cloudera.com. [Dostopano 8. 3. 2015].

[5] A.Simitsis, D. Theodoratos. Data Warehouse Back-End Tools. 2009

[6] Eclipse. [Online]. Dosegljivo: https://eclipse.org/. [Dostopano 6. 3. 2015].

[7] K. Kakish, T.A. Karft. ETL Evolution for Real-Time Data Warehousing. In: Proceedings of the Conference on Information Systems Applied Research. 2012

[8] ETL Tools – Top 10 ETL Tools Reviews [Online]. Dosegljivo: http://www.databaseetl.com/etl-tools-top-10-etl-tools-reviews/. [Dostopano 8. 3. 2015].

[9] R.Katragadda, S. S. Tirumala, D. Nandigam. ETL tools for Data Warehousing: An empirical study of Open Source Talend Studio versus Microsoft SSIS. In: ICWISCE'2015 International Conference on Web Information System and Computing Education, The 2nd World Congress on Computer Applications and Information Systems, 2015

[10] Twitter Support [Online]. Dosegljivo: https://support.twitter.com/articles/131209-faqs-about-top-search-results. [Dostopano 6. 3. 2015].

[11] IBM InfoSphere Information Server [Online]. Dosegljivo: http://www-01.ibm.com/software/data/integration/info_server/ [Dostopano 8. 3. 2015].

[12] Informatica Power Center [Online]. Dosegljivo: https://www.informatica.com/ [Dostopano 8. 3. 2015].

[13] A. Albrecht, F. Naumann. Managing ETL Processes [Online]. Dosegljivo: https://hpi.de/fileadmin/user_upload/fachgebiete/naumann/publications/2008/ETL_Management.pdf. [Dostopano 8. 3. 2015].

[14] MongoDB. [Online]. Dosegljivo: http://www.mongodb.org/.

[15] C. I. Murar Master Thesis: ETL Testing Analyzer. Universitat Politècnica de Catalunya, 2014

[16] Pinterest. [Online]. Dosegljivo: http://www.pinterest.com. [Dostopano 8. 3. 2015].

[17] R. Parida. Principles & Implementation of Datawarehousing. Firewall Media, 2006.

[18] Pentaho Data Integration [Online]. Dosegljivo: http://www.pentaho.com/product/data-integration. [Dostopano 8. 3. 2015].

[19] Talend Big Data Sandbox [Online]. Dosegljivo: http://www.talend.com/solutions/big-data-sandbox. [Dostopano 6. 3. 2015].

[20] SWOT Analysis: Strengths, Weaknesses, Opportunities, and Threats [Online]. Dosegljivo: http://ctb.ku.edu/en/table-of-contents/assessment/assessing-community-needs-and-resources/swot-analysis/main. [Dostopano 8. 3. 2015].

[21] X. Liu, , C. Thomsen, T. B. Pedersen. ETLMR: A Highly Scalable Dimensional ETL Framework Based on MapReduce. In: Transactions on Large-scale Data- and Knowledge-centered Systems VIII: Special Issue on Advances in Data Warehousing and Knowledge Discovery. Springer 2013

[22] Talend ETL software. [Online]. Dosegljivo: http://www.bitool.net/software/talend-open-studio.html. [Dostopano 8. 3. 2015].

[23] Talend [Online]. Dosegljivo: http://www.talend.com. [Dostopano 8. 3. 2015].

[24] Talend Customers [Online]. Dosegljivo: http://www.talend.com/customers. [Dostopano 6. 3. 2015].

[25] Talend Data Integration [Online]. Dosegljivo: http://www.talend.com/download/talend-open-studio#t4. [Dostopano 6. 3. 2015].

[26] Talend Online Documentation & Knowledge Base [Online]. Dosegljivo: https://help.talend.com/display/TalendComponentsReferenceGuide54EN/tMongoDBOutput. [Dostopano 16. 3. 2015].

[27] Kimball, R. Reeves, L., Ross. M., Thornthwaite. The Data Warehouse Lifecycle Toolkit: Export Methods for Designing, Developing and Developing and Deploying Data Warehouses, 1st edition, Indiana: Wiley Publishing Inc. 1998

[28] Twitter Components for Talend [Online]. Dosegljivo: http://gabrielebaldassarre.com/talend/twitter-components-talend/. [Dostopano 6. 3. 2015].